EPC: A Protocol for Dynamic Routing in Sensor Networks with an Emphasis on Energy Conservation

Zach Cooper (CSE undergraduate University of Nevada, Reno)
Roniel Padua (CSE undergraduate University of Nevada, Reno)
Austin Ogilvie (CSE undergraduate University of Nevada, Reno)

Abstract:*Wireless sensor networks (WSNs) are a technology with the a wide variety of potential applications. They rely on spontaneous formation of networks so that sensor data can be transported in a wireless manner. WSN's are built from various nodes where each of those nodes has a sensor that has limited power. Dynamic routing is a network technique which allows the router to select paths based upon the current conditions of the network. In this way, the data is not being transmitted via a fixed path and allows much more malleability for the path of data packets. The two most popular dynamic routing protocols are the Routing Information Protocol (RIP) and the Open Shortest-Path First protocol (OSPF). By using dynamic routing in a WSN, the longevity of the WSN can most certainly be increased assuming the sensors lose some of their finite power every time a packet is transmitted. The team's protocol, Efficient Power Conservation Protocol (EPC), seeks to prolong the lifetime of a sensor network by routing packets in a way to conserve the sensor node's energies for as long as possible.*

Keywords *– WSN, RIP, OSPF****,*** *EPC*

I. Introduction

The team's goal for this project was to develop a routing protocol with an emphasis on energy conservation. To achieve this, we created a sensor network simulator and used our protocol for forwarding packets in the network simulation to show the longevity of the the sensor network via our protocol. Details of the simulation environment can be found in section III. Due to the dynamic routing aspect of the protocol, the packets were forwarded based upon least amount of energy consumption and a minimum power threshold sensor check.

II. Efficient Power Conservation Protocol (EPC)

The team's protocol focuses on conservation of the sensor network to increase the longevity of the sensors. In other words, we wanted to keep the sensors on-line for as long as possible. The protocol decides the paths of least amount of energy consumption to reach a destination hub and will follow that path in order to increase sensor longevity within the network. Sensor energy is decreased after every time a packet is forwarded. Due to the dynamic nature of this protocol, packets are allowed to be

forwarded through virtually any path to their destination hubs in order to conserve energy. The EPC function is essentially a modified version of Dijkstra's algorithm. Rather than follow the shortest path, it follows largest sensor energy.

Error handling is based upon remaining sensor node energy and a minimum power threshold. Once a node's power level reaches a certain threshold, the power conservation protocol re-routes the packets through alternate sensor nodes. This maybe cause the packets take longer to reach their destination, as a packet maybe be forwarded through the network at longer paths as compared to other protocols. However, the sensor nodes will remain active for longer periods of time in this manner. Eventually, since the sensor's power levels are finite, the sensors will stop transmitting once the minimum power threshold is reached. Packet loss was not included in this simulation therefore there was no error handling for it.

A simple example for this routing protocol being used in a sensor network is as follows: a scenario where a packet ***p1*** with a value of 5 energy consumption required per forward ***e*** currently residing in a sensor node ***s1*** with a power level ***pwr*** of 85. The current sensor can see three other sensors ***s2*** and ***s3*** and ***s4***. The sensor ***s2*** has a ***pwr*** of 50 while ***s3*** has one of 90 and ***s4*** has a value of 60. The protocol will send the packet to ***s3*** due to the power level of ***s3*** > ***s2***. ***S1***'s power level will then decrease to 80 because the power consumption required to forward that packet was 5 and ***s1***(85) – ***e***(5) = 80 will be the new value for ***e*** of ***s1***. Once ***s2*** = ***s3***, the packet will randomly choose which sensor to traverse through. However, once a sensor node reaches a certain threshold of minimum power remaining, the packets will then traverse through alternate nodes in order to preserve the life of the sensor node with low power.

III. Simulation Environment Description

The simulation environment was coded in C++ programming language. A modified weighted graph class from the team's previous CS 302 data structures course was used to simulate the structure of the sensor network and packet structure of varying amounts of power consumption to be sent throughout the network nodes, eventually making their way to command and control (C&C) hubs. In order to provide a visual of what is taking place during the simulation, the simulation prints terminal output for the existing edges between sensors, the number of sensors in the network, the number of packets, the power consumption for each of those packers, and the network itself with the sensors and edge weights in a matrix view. The -std=c++11 flag must be used when compiling the project, as there are dependencies required by the code in order for it to compile. Detailed comments and instructions can be found within the code itself.

The team created a simulated sensor network with 20 sensor "nodes" and 5 destination hubs as the destination for 5 packets. Every hop from a packet depletes some of the sensor's power from which it came from. The packet structure includes a simple integer value for the amount of power consumed per hop, a destination vertex and the vertex from which the packet originate. Our protocol is programmed as a function and it loops through the weighted graph. As it loops through the graph, a the protocol will check to see if there is an edge between the other sensors. If an edge between sensors is

found, it will then proceed to check each sensors remaining energy level. The protocol will then choose the sensor with the largest amount of energy left and send the packet to that node. The power consumption integer value will then be subtracted from the sensor that forwarded the packet, resulting in a lower power level for the source node. Below is a generic overview of the simulation environment during the routing of packets:

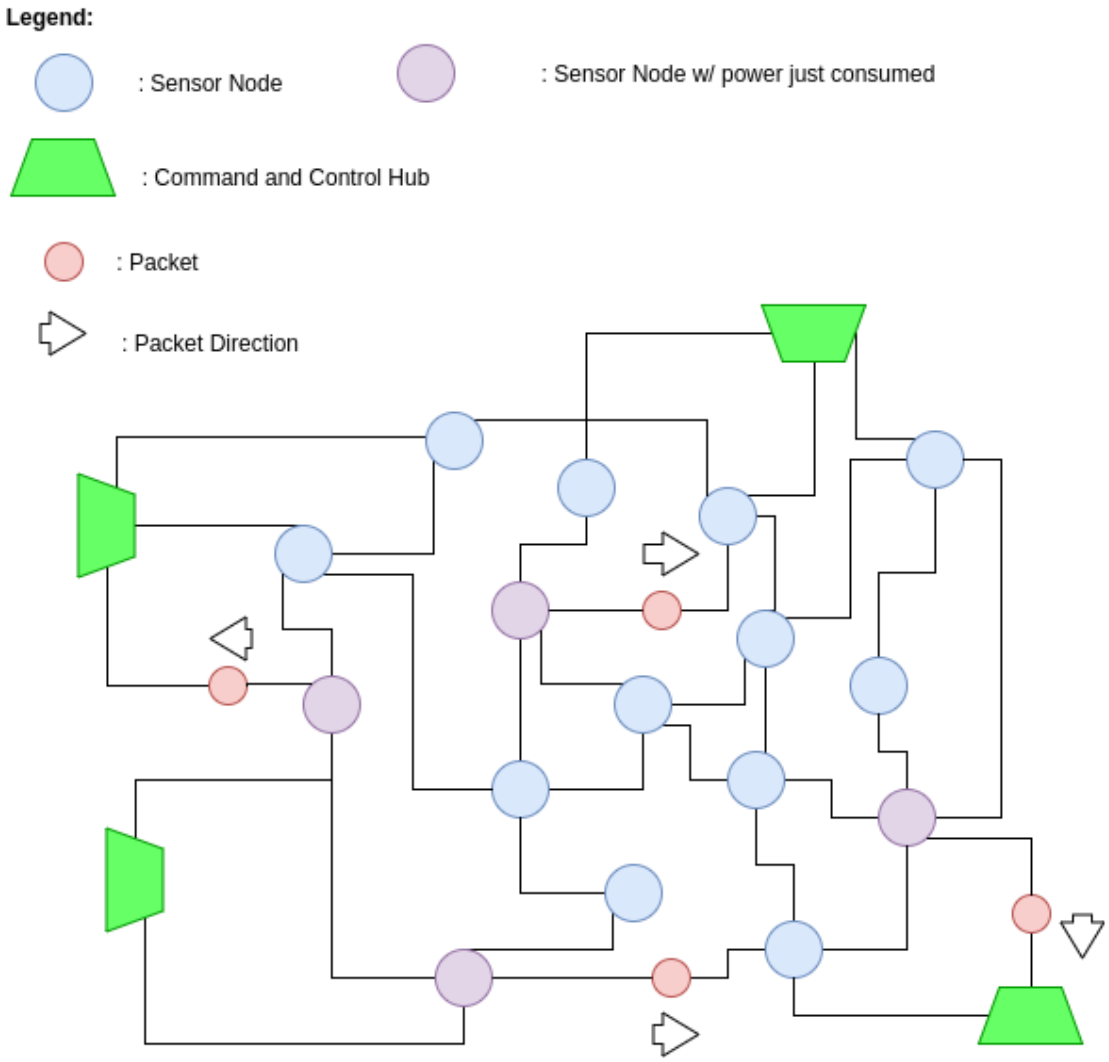


Figure 1: A generic visual of the sensor network forwarding four nodes forwarding four packets at a certain single time frame. The packets colored purple have used up some of their energy by forwarding their respective packets.

In order to increase the complexity of the simulation environment, the C&C destination hubs have been randomized at specified locations and designated as the destination of the packets. The packet source locations have also been randomized. The simulation results are printed to the terminal in order to provide a visual of what is going on. Pictures of the program terminal output have also been included below:

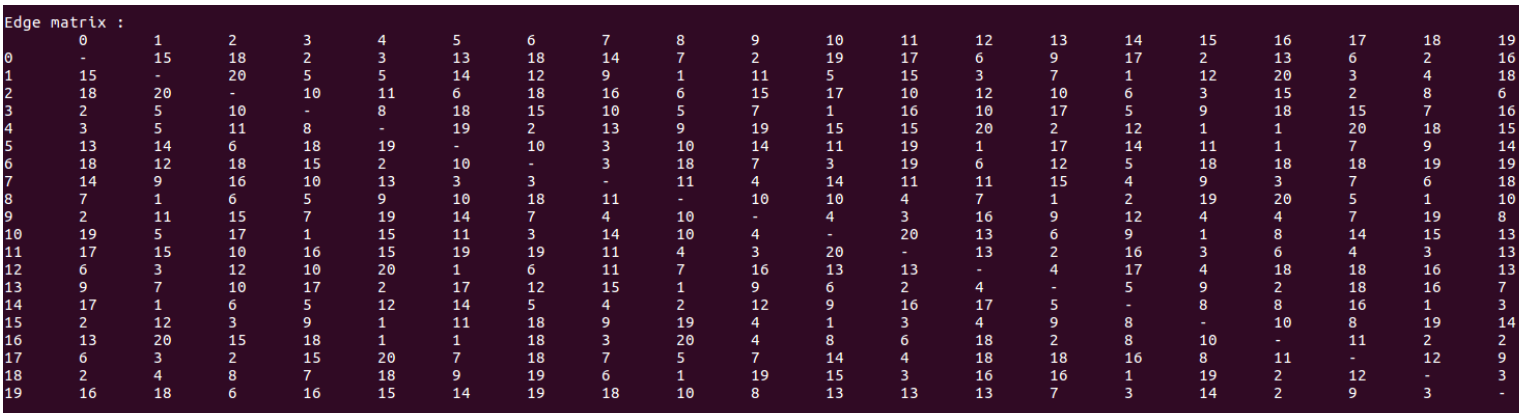
```
Packet #1 energy consumption per packet hop :6
Packet #2 energy consumption per packet hop :7
Packet #3 energy consumption per packet hop :5
Packet #4 energy consumption per packet hop :3

Sensor edge vertex locations:
Size = 20, index of sensor_v1 = 0, index of sensor_v2 = 1
Size = 20, index of sensor_v1 = 0, index of sensor_v2 = 2
Size = 20, index of sensor_v1 = 0, index of sensor_v2 = 3
Size = 20, index of sensor_v1 = 0, index of sensor_v2 = 4
Size = 20, index of sensor_v1 = 0, index of sensor_v2 = 5
Size = 20, index of sensor_v1 = 0, index of sensor_v2 = 6
Size = 20, index of sensor_v1 = 0, index of sensor_v2 = 7
Size = 20, index of sensor_v1 = 0, index of sensor_v2 = 8
```

Figure 2: Terminal output for the packets and their energy consumption values. The beginning of the edge data for the links between the sensors is also included.

The team included various terminal outputs in order to promote easier visuals for the simulation. Below are two more screen-shots of the terminal output from the beginning of the simulation execution.

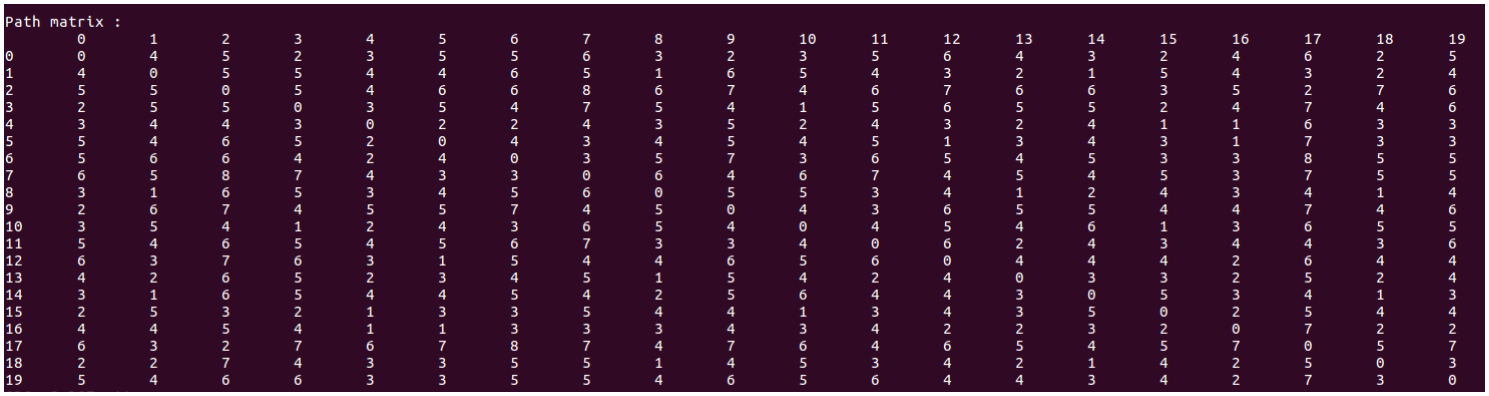
```
Sensor list :
Row     Label   Energy Level
0       0       88
1       1       90
2       2       83
3       3       57
4       4       52
5       5       50
6       6       78
7       7       74
8       8       63
9       9       80
10      10      75
11      11      87
12      12      59
13      13      84
14      14      71
15      15      88
16      16      92
17      17      51
18      18      78
19      19      72
```

Figure 3: Terminal output for the all the sensors in the network, including their row location and starting energy level.



```
Edge matrix :
        0       1       2       3       4       5       6       7       8       9       10      11      12      13      14      15      16      17      18      19
0       -       15      18      2       3       13      18      14      7       2       19      17      6       9       17      2       13      6       2       16
1       15      -       20      5       5       14      12      9       1       11      5       15      3       7       1       12      20      3       4       18
2       18      20      -       10      11      6       18      16      6       15      17      10      12      10      6       3       15      2       8       6
3       2       5       10      -       8       18      15      10      5       7       1       16      10      17      5       9       18      15      7       16
4       3       5       11      8       -       19      2       13      9       19      15      15      20      2       12      1       1       20      18      15
5       13      14      6       18      19      -       10      3       10      14      11      19      1       17      14      11      1       7       9       14
6       18      12      18      15      2       10      -       3       18      7       3       19      6       12      5       18      18      18      19      19
7       14      9       16      10      13      3       3       -       11      4       14      11      11      15      4       9       3       7       6       18
8       7       1       6       5       9       10      18      11      -       10      10      4       7       1       2       19      20      5       1       10
9       2       11      15      7       19      14      7       4       10      -       4       3       16      9       12      4       4       7       19      8
10      19      5       17      1       15      11      3       14      10      4       -       20      13      6       9       1       8       14      15      13
11      17      15      10      16      15      19      19      11      4       3       20      -       13      2       16      3       6       4       3       13
12      6       3       12      10      20      1       6       11      7       16      13      13      -       4       17      4       18      18      16      13
13      9       7       10      17      2       17      12      15      1       9       6       2       4       -       5       9       2       18      16      7
14      17      1       6       5       12      14      5       4       2       12      9       16      17      5       -       8       8       16      1       3
15      2       12      3       9       1       11      18      9       19      4       1       3       4       9       8       -       10      8       19      14
16      13      20      15      18      1       1       18      3       20      4       8       6       18      2       8       10      -       11      2       2
17      6       3       2       15      20      7       18      7       5       7       14      4       18      18      16      8       11      -       12      9
18      2       4       8       7       18      9       19      6       1       19      15      3       16      16      1       19      2       12      -       3
19      16      18      6       16      15      14      19      18      10      8       13      13      13      7       3       14      2       9       3       -
```

Figure 4: Terminal output for the graphical matrix view of the sensor network. This includes, rows, columns and edges for the sensors in the network.



```
Path matrix :
        0       1       2       3       4       5       6       7       8       9       10      11      12      13      14      15      16      17      18      19
0       0       4       5       2       3       5       5       6       3       2       3       5       6       4       3       2       4       6       2       5
1       4       0       5       5       4       4       6       5       1       6       5       4       3       2       1       5       4       3       2       4
2       5       5       0       5       4       6       6       8       6       7       4       6       7       6       6       3       5       2       7       6
3       2       5       5       0       3       5       4       7       5       4       1       5       6       5       5       2       4       7       4       6
4       3       4       4       3       0       2       2       4       3       5       2       4       3       2       4       1       1       6       3       3
5       5       4       6       5       2       0       4       3       4       5       4       5       1       3       4       3       1       7       3       3
6       5       6       6       4       2       4       0       3       5       7       3       6       5       4       5       3       3       8       5       5
7       6       5       8       7       4       3       3       0       6       4       6       7       4       5       4       5       3       7       5       5
8       3       1       6       5       3       4       5       6       0       5       5       3       4       1       2       4       3       4       1       4
9       2       6       7       4       5       5       7       4       5       0       4       3       6       5       5       4       4       7       4       6
10      3       5       4       1       2       4       3       6       5       4       0       4       5       4       6       1       3       6       5       5
11      5       4       6       5       4       5       6       7       3       3       4       0       6       2       4       3       4       4       3       6
12      6       3       7       6       3       1       5       4       4       6       5       6       0       4       4       4       2       6       4       4
13      4       2       6       5       2       3       4       5       1       5       4       2       4       0       3       3       2       5       2       4
14      3       1       6       5       4       4       5       4       2       5       6       4       4       3       0       5       3       4       1       3
15      2       5       3       2       1       3       3       5       4       4       1       3       4       3       5       0       2       5       4       4
16      4       4       5       4       1       1       3       3       3       4       3       4       2       2       3       2       0       7       2       2
17      6       3       2       7       6       7       8       7       4       7       6       4       6       5       4       5       7       0       5       7
18      2       2       7       4       3       3       5       5       1       4       5       3       4       2       1       4       2       5       0       3
19      5       4       6       6       3       3       5       5       4       6       5       6       4       4       3       4       2       7       3       0
```

Figure 5: Terminal output for shortest path algorithm being run on the sensor network. This includes, rows, columns and edges and the shortest paths for packet source to destination for the network.

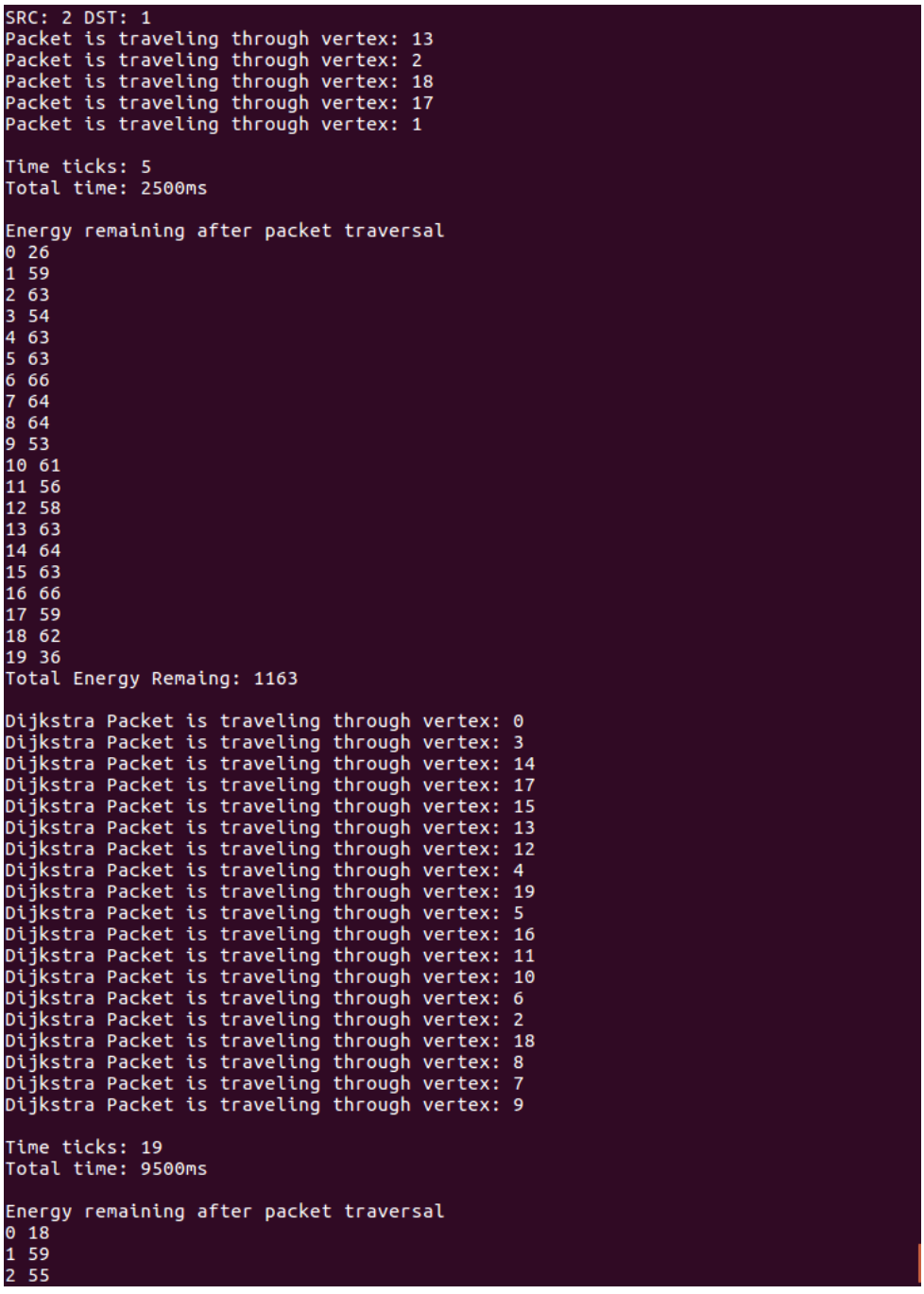
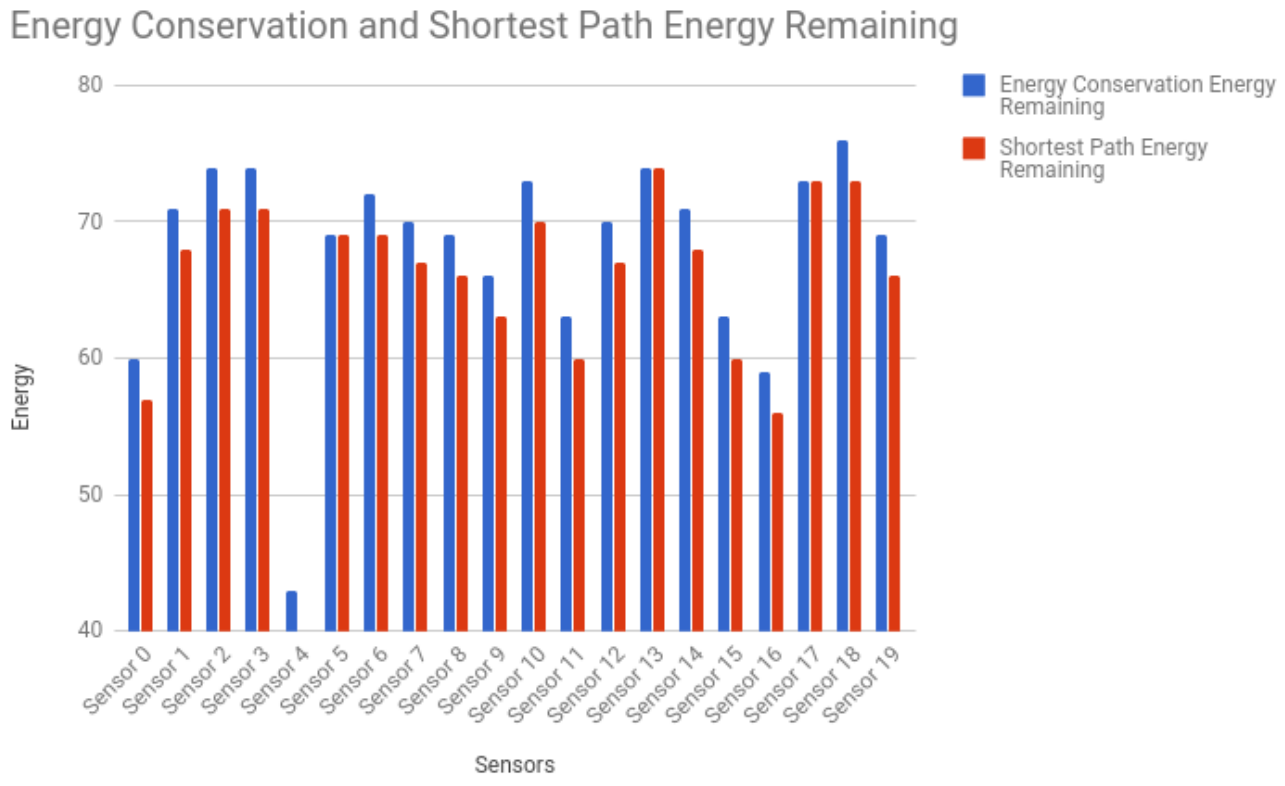


Figure 6: Terminal output of the packet paths throughout the network, selecting the paths of least amount of energy consumed.

IV. Simulation EPC Results vs. Other Protocol

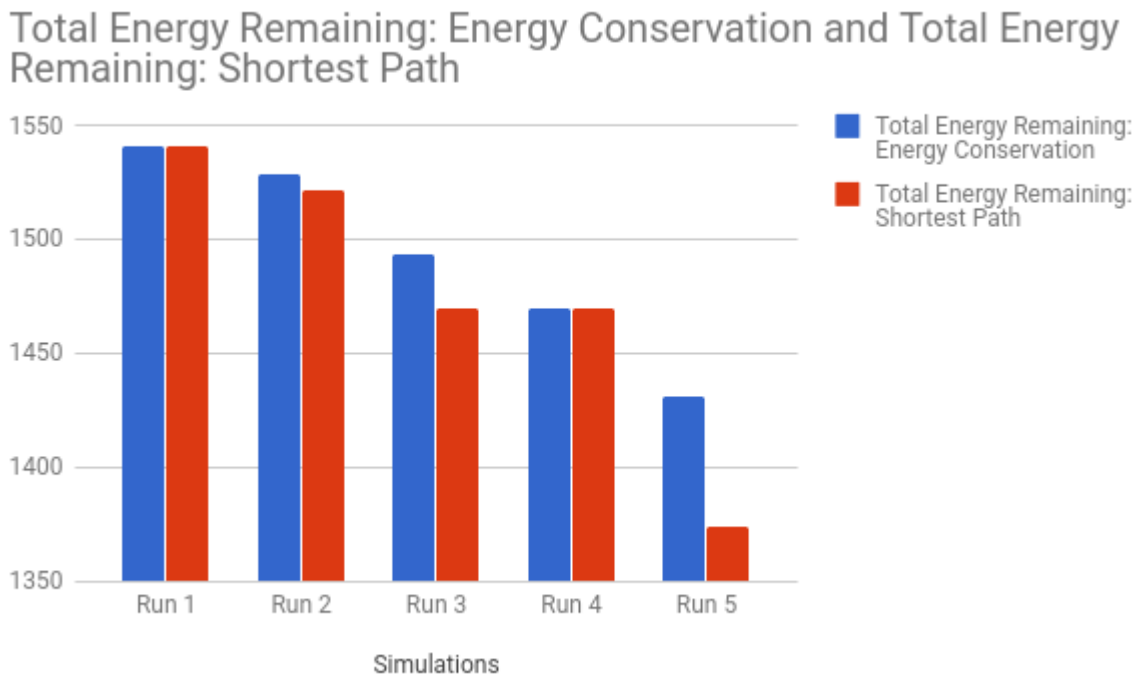
Below, we have included some results via graphical representation for our power conservation protocol vs. a shortest path algorithm. The shortest path algorithm is meant to simulate the OSPF protocol in the sensor network. The shortest path algorithm traversed the sensor network via the shortest path based on the edge weights of the graph. It completely ignores the energy level check and will consume sensor energy at a much quicker rate than the team's EPC protocol. As shown below, the sensor network was able to increase its longevity much more with the power conservation protocol vs. the shortest path. The shortest path was able to deliver the packets more quickly, however various sensor nodes power was consumed at a much quicker rate causing the network to power down more quickly.



Graph 1: A bar graph showing the amount of energy remaining in the each of the 20 individual sensors after the packets have reached their destination from a single run. The blue bars represent the amount of energy left after the packets traverse the network via the team's EPC protocol and the red demonstrates the energy remaining in the sensors from a shortest path protocol.

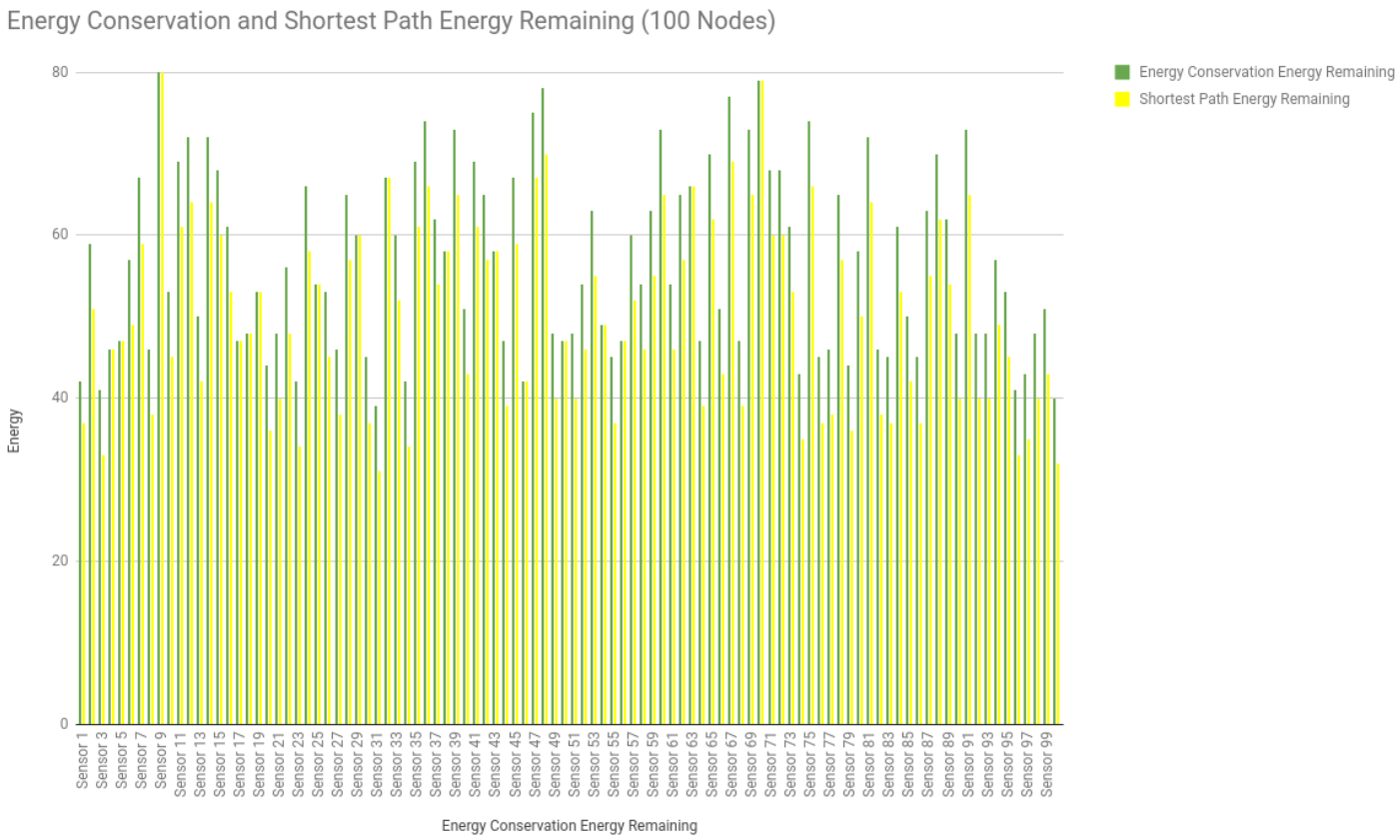
In general, the team's energy conservation protocol was able to retain sensor energy most of the time. However, in some cases the shortest path used about the same amount of energy due to the

randomization of the network parameters in the form of packets energy consumption and sensor power levels.

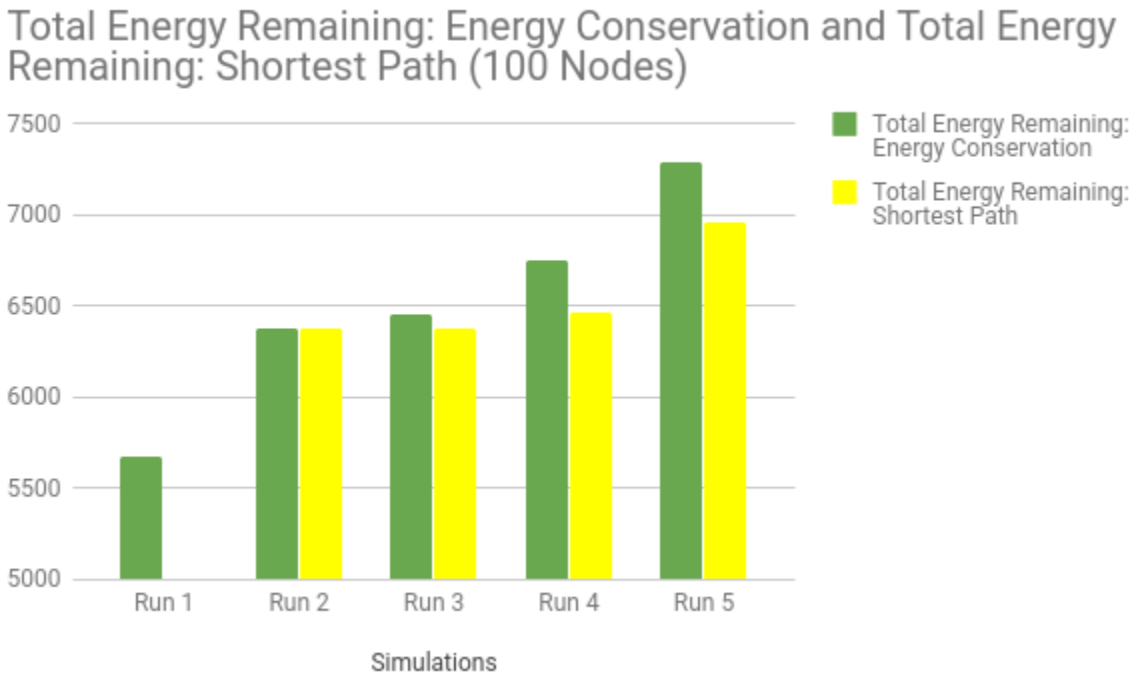


Graph 2: Energy remaining in total for all the collective sensors in the 20 sensor network after the five packets have reached their destinations. Five simulation runs were taken in this case. Blue is the team's EPC protocol while red is the shortest path. By run 5, it is clear the energy is decreasing at a far quicker rate in the shortest path remaining traversal.

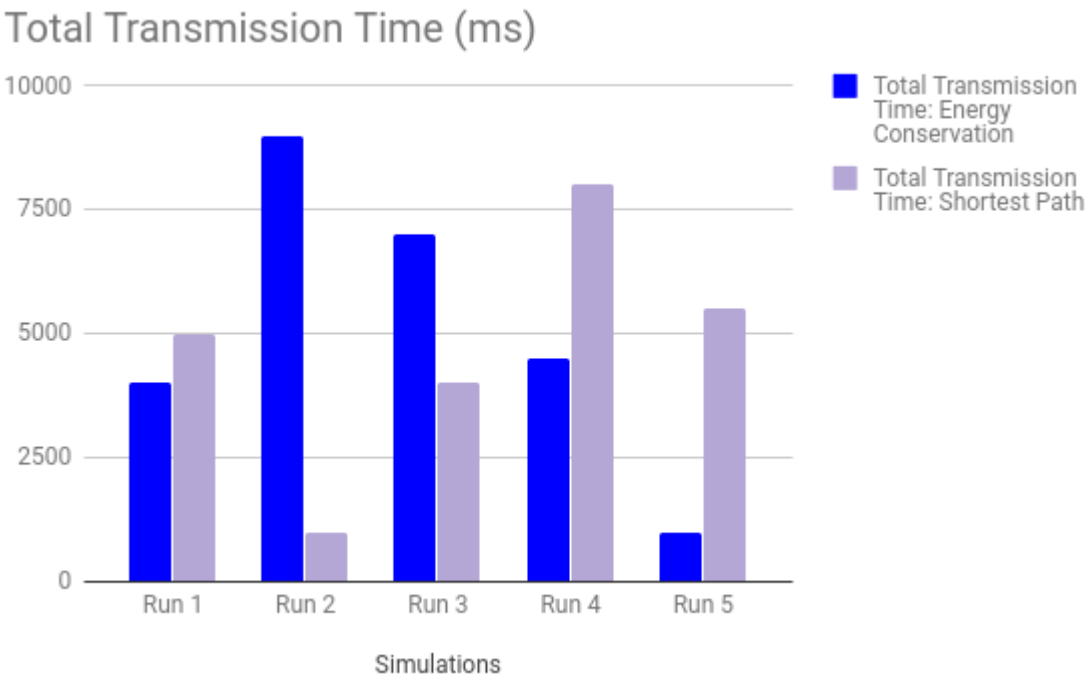
The team also decided to graph the results of many sensors. A 100 sensor node network with 5 packet configuration was used. To attain a more realistic view or our protocol in the sensor node network, the EPC and shortest path protocols were once again ran 5 times. From graphical results of the 100 sensor node network seen below, it is obvious EPC is a much more beneficial way to prolong sensor energy.



Graph 3: A bar graph showing the amount of energy remaining in the each of the 100 individual sensors after the packets have reached their destination from a single run. The green bars represent the amount of energy left after the packets traverse the network via the team's EPC protocol and the yellow demonstrates the energy remaining in the sensors from a shortest path protocol.



Graph 4: Energy remaining in total for all the collective sensors in the 20 sensor network after the five packets have reached their destinations. Five simulation runs were taken in this case. Green is the team's EPC protocol while yellow is the shortest path. By run 5, it is clear the energy is decreasing at a far quicker rate in the shortest path remaining traversal.



Graph 5: The time taken for the packets to be transmitted from their sources to their destinations. The blue represents EPC while the gray represents the shortest path.

V. Conclusion and Possible Future Improvements

In conclusion, our EPC protocol is ideal for sensor networks due to the increased longevity it provides for the sensors as a whole. As seen from the analysis above, the EPC protocol increased the longevity of the sensor's in the sensor network. The team also measured the amount of time the network remained online and it was clear by the time graph, the shortest path killed the sensor network at a quicker rate than EPC. Future work would include account for errors such as packet loss or packet delay when transmitting. More error handling would be included to deal with these issues and it would make the overall simulation much more realistic.

References

[1] "*ICT - Energy - Concepts Towards Zero - Power Information and Communication Technology*", *book edited by Giorgos Fagas, Luca Gammaitoni, Douglas Paul and Gabriel Abadal Berini, ISBN 978-953-51-1218-1, Published: February 12, 2014 under CC BY 3.0 license. © The Author(s).*

[2] Arati Majeshwar, Dharma P. Argrawal, "*TEEN: A Routing Protocol for Enhanced Efficiency in Wireless Sensor Networks**" , Proc. of IEEE 2001